1.0 Introduction

1.1 Purpose of SRS
The purpose of this SRS is to outline the design for an Electronic Voting system. The E-Voting system is a concept that many companies have tried to develop and deploy to the public and private sector. However these systems have been lacking in areas such as security, anonymity and audibility. Our goal is to develop a system that will satisfactorily meet these standards.

1.2 Scope of Product
The scope of the E-Voting system that we are specifying is for elections of up to 20,000 people. Specifically, we will design a system for the Student Association elections here at Syracuse University.

1.3 Acronym
1.3.1 DRE: Direct Recording Electronic.
1.3.2 SA: Student Association.
1.3.3 CMS: Computing and Media Services.
1.3.4 SSH: Secure Shell.
1.3.5 GUI: Graphical User Interface.
1.3.7 TCP: Transmission Control Protocol.

1.4 References:
1.4.1. www.acm.org/usacm/issues/evoting.htm
1.4.2. www.avirubin.com/vote
1.4.3. Student Association code of statutes
1.4.4. http://students.syr.edu/sa/

1.5 Overview of the Rest of SRS
The rest of this document will specifically cover what would be needed to design an E-Voting system considering standards such as integrity, security, usability, anonymity, scalability, speed, audibility and accuracy. The two primary standards that should be stressed are security and audibility. The shortcomings of many other systems are their inability to provide sufficient security for voters or an audit trail to verify the votes that were cast.

2.0 General Description

2.1 Product Perspective
The SA at Syracuse University gives students the opportunity to contribute to the academic and social community. All of the members of the SA are elected to their
respective positions by means of a traditional, democratic election. In the past, the election was conducted by means of paper ballots. But as technology becomes more prevalent in society, tedious operations like conducting and tallying a paper ballot voting system become obsolete.

Electronic voting is something that is in hot debate on the national level. With the recent troubles that have come with certain paper ballot systems in recent elections, many feel that a secure and reliable electronic voting system could prevent such problems from occurring in the future. However, whether or not a national system that is secure and reliable is possible is yet to be determined. But on a smaller scale, such as a general student body election at a relatively small private university, security and reliability measures are not as difficult to implement.

The SA has attempted to take advantage of new online voting technology in recent years, hiring someone who designed and successfully implemented an online system. This system has several holes, however, which we intend to fix in our design. The aim of our design is to maximize simplicity of use on the side of the voter, as well as the administrators. This will be accomplished effectively if we focus on the flaws of the current online system and fill in the missing pieces.

2.2 Product Functions
The software's main function is to conduct an election of officials using the internet to allow people to vote. The functions of the systems relate specifically either to the process of a person casting a vote or an administrator running the election.

A student who would like to cast a vote should first be directed through the school’s website to the election's webpage. Here, he/she will be asked to log in to the system. The system should verify that the student has not voted yet and then bring them to an instruction page. After the student accepts the terms of the votes and understands how to vote, he/she will be brought to the voting page. This page should list the positions and candidates. After the student has selected the candidates he/she would like to vote for, the system should check the validity of the vote and bring up a confirmation page. After the student confirms his/her vote, he/she will be able to print out and save a receipt.

Once the vote confirmed, the vote will be transmitted over the internet to a central server. The information recorded will include the time of the vote, candidates voted for and an id number unique to the student. A hashing function will use this information to generate a unique confirmation number for the user. The votes for the candidates will be tabulated and the vote information will be printed out, available for the student to verify his/her vote.

The software should also have an interface for administrators to allow a new election to be set up and a current election to be edited if problems arise.

2.3 User Characteristics
The users of the E-voting system will be students enrolled at Syracuse University. The diverse student body consists of a wide variety of backgrounds and specializations. Nearly all of these students have had plenty of exposure to computers throughout their educations and have basic computer skills. However, the possibility that some students do not possess such skills must be considered. The system design should be user-friendly enough that all students will be able to use the system with minimal instruction. Furthermore, help must still be made available during the election to insure that these students will be able to successfully place their votes with the same confidence that their vote has been tallied correctly as everyone else.

There are also students at the university with a wide range of disabilities. These students must be given the same right to vote confidently as everyone else. Since the system might cause trouble for students with certain disabilities, the help of Disability Services should be recruited to guarantee students the necessary assistance that they might require. Directions to these sources of assistance should be made readily available to those with disabilities to lighten the work to be done by these already burdened students.

2.4 General Constraints
The main system hardware constraint will be the server’s ability to hold the election data. Though the election is limited to 20,000 voters, confidence in the server’s ability to accurately contain the information and remain functional throughout the election is essential. Any problems in this area could be devastating to the election process.

The interface must be simple to navigate and allow the user to successfully check off their votes. Voters should be provided with adequate instruction while using the system. The system must also be able to handle write-in votes, where a student database will need to be accessed and checked. The interface should be fairly plain in order to keep the focus of the voter on the task at hand. No form of candidate campaigning will be allowed to be a part of the interface to ensure that the voter is not unfairly persuaded.

The software will be web-based and will need to function on a variety of operating systems to allow voters as widespread access as possible.

Because the election will be web-based, vote verification by user is difficult. Ideally, the user could have their voted printed right before them to verify and turn in. However, when the votes are tabulated at a separate location, this is not possible. Instead, the system should allow the user to verify their vote at this separate location using the paper trail.

The database and the functions that use it must be designed without storing information matching students with their respective votes. It must not be possible for anyone to use the system to check who a specific student voted for. The anonymity of the voting process must be maintained.

2.5 Assumptions and Dependencies
The assistance of CMS is depended on for the election’s login process. The software should allow students to log in to the system using their CMS login names and passwords. Access to this data must be provided by the school.

When a voter attempts to make a write-in vote, they should be able to search for and select the student’s name from a list. Thus, the school must provide a list of the full names of all active students, along with class and home college information.

Vote transfer is dependent on access to fast and reliable servers on the campus, along with constant access to the internet for students throughout the election. Storage of the election data depends on access to a large enough database to hold all of the vote information.

The assistance of Disability Services is depended on for helping students whose disabilities hinder their ability to cast their votes successfully and confidently.

The SA is depended on to provide information on the nuances of the election. The SA is assumed to continue to be interested in the development of the system and to be present throughout its implementation in order to insure that their standards are being met.

The security of the system as a whole depends on reliable coworkers. Those overseeing the main terminal and the paper trail will need to protect the election information. Other people who have access to security information are assumed to be trustworthy enough not to leak information to possible attackers.

3.0 Specific Requirements

3.1 Functional Requirements

3.1.1 Display Instructions

3.1.1.1 Specification

3.1.1.1.1 Introduction
The purpose of the instructional page is to give the user clear directions on how to use the voting system. This should include a disclaimer summarizing the penalty of unauthorized or fraudulent system access as outlined in the SU Computing and Electronic Communications Policy. The page should also contain information on the confidentiality and security of the user’s vote.

3.1.1.2 Inputs
After reading the instructional page, the user will be asked whether they agree to the terms of the disclaimer. They will inform the system of their decision by choosing between an “I agree” and “I disagree” button located at the bottom of the page.

3.1.1.3 Processing
Instructions on how to cast a vote should be displayed clearly to the user, followed by two buttons. If the user clicks on the "I disagree" button they are directed back to the SA site, and if they click on the "I agree" button then they are sent to the voter login page.

3.1.1.4 Outputs
The outputs for this requirement should be the redirection of the user to the login page or the SA home page depending on whether they agreed to the disclaimer or not.

3.1.2 External interfaces

3.1.2.1 User Interfaces
The user will see a title followed by instructions on how to use the voting system, which should be written plainly and very clearly. There will be a disclaimer describing the legal aspects of voter fraud. The voter must understand that voting under an identity that is not their own is a breach of this contract, as is any other tampering with the results of the election. The user should also be assured that their vote will be processed securely and in confidentiality. Two buttons labeled “I agree” and “I disagree” should be located at the bottom of the page, to allow the user to confirm that they have read and understand the instructions and disclaimer and are prepared to cast their vote.

3.1.2 Login

3.1.2.1 Specification

3.1.2.1.1 Introduction
The program should allow the user to enter their NetID and password supplied by CMS in order to log in to the voting system. Once entered, the information will be transmitted to the CMS LDAP server for confirmation. The login server will indicate whether or not the login attempt was successful. If the login fails the user will be returned to the login screen with an appropriate error message. If login is successful then the system should check to see if the user has already voted. If the user has voted then they are taken to a page that explains that they have previously voted. Otherwise the user is taken to the vote page.

3.1.2.1.2 Inputs
The inputs for this function are the NetID and password. The user should already have knowledge of his/her NetID and password, but in case they do not there should be a message telling them to contact CMS.

3.1.2.1.3 Processing
Once the user enters their NetID and password, the information is sent to the login server for verification. The login server will be supplied by CMS. The login server will then return either a successful login or login failed and the student’s home college in order to direct the user to the appropriate page.

3.1.2.1.4 Outputs
The outputs for this function will be successful or unsuccessful login, which will then direct the user to the appropriate page.

3.1.2.2 External Interfaces

3.1.2.2.1 User Interfaces
The user interface will be a page with two text boxes, labeled username and password. Below those should be a button that will begin the login verification process once clicked.

3.1.2.2.2 Software Interfaces
The voting software will interact with the software of CMS with regard to user verification and information retrieval.

3.1.2.2.3 Communication Interfaces
Pending a response from CMS, the parameters of the communication interface are unknown at this moment. Presumably, the client's NetID and password should be sent using a secure protocol, I.E. stunnel or whatever protocol CMS recommends.

3.1.3 Multiple Vote Restriction

3.1.3.1 Specification

3.1.3.1.1 Introduction
When a user who has already voted attempts to login to the system again, they will be informed that they have already voted and that they may not advance any further.

3.1.3.1.2 Inputs
This function will take receive all system login attempts for processing.

3.1.3.1.3 Processing
The system should send the information for the login attempt over the server to the main database where it can check whether or on the use has voted or not. When the system receives an attempt to log into the system with a user name that has already voted, it should direct the user to a page indicating this and deny their login attempt.

3.1.3.1.4 Outputs
The user should be shown a message explaining that they have already voted through the system and cannot vote again. The user should be reminded that they can check the vote that the system registered for them by bringing their receipt to the appropriate SA election official, where the paper trail will be available.

3.1.3.2 External Interfaces

3.1.3.2.1 User Interfaces
The user should be shown the information already discussed.
3.1.3.2.2 Communication Interfaces
The voting application will transmit the user's login name to the main database, which will return true if the user has voted and false if the user has not, or any similar method of recognizing a vote has already been recorded.

3.1.4 Display Ballot

3.1.4.1 Specification

3.1.4.1.1 Introduction
The user should be brought to a new page after they click the “I agree” button on the instructions page. The page will contain a list of the positions and candidates, with a box to check next to each one, which the user will use to cast a vote for that candidate. The page will be unique to the student’s home college at the university because each college must elect “assemblymen”, for whom only students enrolled in that college can vote for. There should also be a space for write-in votes, which will be discussed later in this document. At the bottom of the page, there should be a button for the user to click to submit his/her ballot. The user should be instructed to click this button only once so as not to cause problems if there is a delay in the system’s response.

3.1.4.1.2 Inputs
The application will use the student’s home college and the candidate information loaded by an administrator to display the correct ballot for that student. The user inputs on this page will be the vote for each position, in the form of checked and unchecked boxes. If the user would like to vote for a write-in candidate, he/she should enter the name of the person into a textbox and perform a search. The user should select the correct name from the entries found and then check the corresponding box.

3.1.4.1.3 Processing
Once the system receives the series of checked and unchecked boxes, it should check to make sure that they correspond to a valid vote, (i.e. two votes cast for two presidential candidates is invalid.) If there is a problem, the user should be made aware of this and be instructed to fix the problem. If the vote is valid, the user should be brought to a confirmation page.

When the user performs a search in order to vote for a write-in candidate, the application should send the search query over the server to the database where it can be checked against a list of students enrolled at the university. The returned list should include the students’ full names along with class status and school enrolled in, to ensure that the user can identify the right person.

3.1.4.1.4 Outputs
The output after the vote has been submitted will be a confirmation page, displaying the vote and asking the user to confirm and submit their vote.
3.1.4.2 External interfaces

3.1.4.2.1 User interfaces
The interface should separate each position to be voted for in the election into a different section of the voting page. Under a title declaring the position should be a list of the candidates running for that position, along with a check box next to each name which the user will use to select to vote for that candidate. For each position, there should also be a spot for a write-in vote and a check box next to that as well. The organization of this page is crucial in order to guarantee that the user can complete their vote successfully. The data should be simple in presentation and there should be no confusion as to how to vote for a candidate. The user should be given the option to clear the entire page of their input using a button at the bottom. There must also be a button to submit the input.

3.1.4.2.2 Communication Interfaces
During a search for a write-in candidate, the voting application will transmit the user's search query to the main database, which will return an information package across the server for each possible candidate, which will include their full name, class, and home school.

3.1.5 Confirm Vote

3.1.5.1 Specification

3.1.5.1.1 Introduction
The confirmation page should allow the user to confirm that the vote being sent into the system is what they intended. The window should show all of the candidates voted for by the individual. The user should be given the option of going back and changing any of their votes. Once they are satisfied, they can officially submit their ballot by clicking the appropriate button.

3.1.5.1.2 Inputs
The only room for user input on this page will be two buttons on the bottom of the page. One button lets the user go back and change their vote, and the other button submits the vote.

3.1.5.1.3 Processing
By clicking the “Go Back” button, the user will be brought back to the voting page and allowed to change any vote they wish to. Clicking the “Submit” button sends the votes to the server and brings the user to the “Receipt” page. After the ballot has been officially submitted, the user can no longer change their vote.

3.1.5.1.4 Outputs
After clicking to submit their ballot, the vote data should be sent to the main database over the server and the user should be directed toward a receipt page. If the user clicks to return to their ballot, the exact voting page that they were previously at should be shown.
This page will reflect the votes as previously understood by the system, and will be completely editable.

3.1.5.2 External Interfaces

3.1.5.2.1 User Interfaces
The page will have each category listed clearly, with the selected candidate listed underneath its respective heading. At the very bottom of the page the user should see the “Go Back” and “Submit” buttons.

3.1.5.2.2 Communication Interfaces
When the user confirms their vote, the application should send the information over the server to the main database to be recorded.

3.1.6 Issue Receipt

3.1.6.1 Specification

3.1.6.1.1 Introduction
The Receipt page is the key aspect in user verifiability and the maintenance of a paper trail. This page is generated after the vote has been submitted, and allows the user to print or save a copy of their vote.

3.1.6.1.2 Inputs
The application should use the vote information that was just collected. Also, the user should be given the option to save or print their receipt, and the option to log out of the system once they are satisfied.

3.1.6.1.3 Processing
The vote information should be reproduced for the user to see. The application should also produce an id number unique to each user and a confirmation number, which should be generated as a function of the time, unique id, and candidates voted for. The unique id number must be unique to all users. The confirmation number does not necessarily have to be perfectly unique, but cannot have too many overlaps. One duplicate per thousand users would be acceptable. When the user clicks the print button, the receipt is sent to the printer queue to be printed. Clicking save allows the user to save a copy of the receipt somewhere to their hard drive. Choosing to logout ends the user’s current interaction with the system and closes the window.

3.1.6.1.4 Outputs
The receipt should be printed to the screen with the information listed above for the user to review and save. If the user chose to print, they should receive a printed copy of the receipt displayed on the screen, via the printer they selected.

3.1.6.2 External Interfaces
3.1.6.2.1 User Interfaces
The page will have each category listed clearly, with the selected candidate listed underneath its respective heading. At the very bottom of the page the user will see the “Print”, “Save” and “Submit” buttons.

3.1.6.2.2 Hardware Interfaces
The computer being used must communicate with a printer in order for the print option to be executable. The save button will need to communicate with the system to save a copy of the vote.

3.1.7 Update Database

3.1.7.1 Specification

3.1.7.1.1 Introduction
When a vote is submitted, the main database that contains all of the information concerning the current state of the election should be updated to reflect the new data.

3.1.7.1.2 Inputs
A vote will be received through the server.

3.1.7.1.3 Processing
The database should record the vote information. The voter’s unique id number, candidates voted for, time voted, and the confirmation number should be stored. All pertinent counters, such as total votes and votes received by each candidate should be incremented.

3.1.7.1.4 Outputs
A fully updated database reflecting all current information should be available.

3.1.8 Paper Trail

3.1.8.1 Specification

3.1.8.1.1 Introduction
One of the major complaints with current E-voting systems is the absence of a verifiable paper trail. The term “paper trail” refers to a sheet of paper that has the vote information printed on it. The purpose of this sheet of paper is so that the voter can confirm that the system has correctly recorded their vote. The voter will be able to do this by bringing the unique identification number and confirmation number that they are provided with after confirming their vote to the station where the paper trail is accumulating.

3.1.8.1.2 Inputs
When the application sends the vote information to the main terminal, the function to produce the paper trail will receive the vote information along with the unique id created for that corresponding voter, timestamp, and confirmation number.

3.1.8.1.3 Processing
The function will print out the information listed above to preferably a dot matrix printer so that each vote can be printed immediately after its reception. If no such printer is available or compatible, the SA can customize the exact algorithm for printing the paper trail. We suggest that pages should be printed when one of two things occurs:
1. The terminal has received enough new votes to fill a page of paper.
2. A period of time, e.g. 30 minutes, has passed without a page being printed.

3.1.8.1.4 Outputs
The printer will print out an entry including the vote information, time, voter’s unique id number, and confirmation number to the ongoing trail. See the “Processing” section (3.1.8.1.3) for details on the page layout.

3.1.8.2 External Interfaces
3.1.8.2.1 Hardware Interfaces
The hardware will include a printer to record the vote information, preferably a dot matrix printer or one that can print a few lines at a time without using a full page. If not, a normal printer will suffice.

3.1.8.2.2 Communications Interfaces
A paper trail line will need to be recorded every time that the system receives a vote from the server.

3.1.9 Vote Verification
3.1.9.1 Specification
3.1.9.1.1 Introduction
The system must give voters the ability to verify that their vote has been recorded correctly by the computer. An online election creates problems in this respect as voters cannot directly print something to hand in as their vote. The voter cannot be completely sure that the vote that they confirm is the vote sent to the system. In order to reduce this questionability, the paper trail has been instituted. A standard must exist for users to verify their votes against this paper trail. Users must have their unique id number in order to access and view their paper trail page. To rightfully declare that their vote has been incorrectly recorded, they must have a copy of their receipt containing all relevant information.

3.1.9.1.2 Inputs
Users wishing to verify their vote should bring their printed receipt to the location where the paper trail is accumulating.
3.1.9.1.3 Processing
When someone presents their unique id number, an election official will present them the vote information recorded by the paper trail. If there is a dispute over the accuracy of the recorded vote, the user should submit their receipt to be paired with the paper trail document including their vote for review by the SA.

The following rules are a suggestion for how the SA could handle such situations.
1. If the timestamp is incorrect, it is likely the vote receipt has been forged so the vote should be kept as previously recorded.
2. If the timestamp is correct,
   (a) If the confirmation number is the same, but the vote information is different, check to see if the confirmation number is correct given the voter’s information using the confirmation number calculator. If it is incorrect on the voter’s receipt, it is very likely the receipt has been forged, so the vote should be kept as previously recorded.
   (b) If the confirmation number is different, check to see if it is correct given the voter’s receipt information or correct with the paper trail’s information. If it is correct, consider changing the vote. If not, the document is likely forged, so the vote should be kept.

3.1.9.1.4 Outputs
Votes should be changed if necessary. Otherwise, users should simply leave with confidence in the accuracy of their vote.

3.1.9.2 External Interface

3.1.9.2.1 User Interfaces
The user will verify their vote by bringing their receipt to the specified location of the paper trail. Election workers will be there to help users verify their votes and deal with any questions.

3.1.9.2.2 Software Interfaces
If there is a complaint of an incorrectly recorded vote and a discrepancy in confirmation number, the SA will want to check the confirmation number by entering the vote information into the confirmation number calculator, made available as a stand alone application.

3.1.10 Turning System ON/OFF

3.1.10.1 Specification

3.1.10.1.1 Introduction
One of the key features in holding a maintainable election is the ability to open and close the polls. The voting system must be able to control voter access by allowing administrators to turn the system on and off for public use. This needs to be done at the beginning of the election period, the end of the election, and any scheduled down time in between. This process should require little to no effort.
3.1.10.1.2 Inputs
The input would be to change the state of the system as it currently stood. The only two possible states to be considered are ON and OFF.

3.1.10.1.3 Processing
If the system is told to turn on, the system will allow users to login and cast votes. If the system is told to shutoff, the system will no longer allow users to login and cast votes.

3.1.10.1.4 Outputs
The new status of the system must be displayed to indicate a successful operation.

3.1.10.2 External Interfaces

3.1.10.2.1 User Interfaces
This administrator function, as well as all administrator functions, is accessible only at the terminal adjacent to the server. A GUI interface will be developed to allow ease in maintenance.

3.1.10.2.2 Software Interfaces
The voting software should be shut down when the system is turned off. Attempts to log in to the system using the online application should be met with a message explaining why votes are not being accepted at that time

3.1.11 Election Setup

3.1.11.1 Specification

3.1.11.1.1 Introduction
From the main menu of administrator functions, there must be a way to setup a new election each year. New elections require both a fresh database to store votes, and a new interface as viewed by the user.

3.1.11.1.2 Inputs
For inputting new candidates, there needs to be a simple GUI that will accept new names under their respective categories. The GUI need not resemble the voter interface, as formatting is to be handled internally.

3.1.11.1.3 Processing
New candidate names will be stored in the system to be formatted for display during the voting process. New databases need to be created to hold the new candidates also. System design should handle matching candidates to the database with no need for the administrator to ever look at code.

3.1.11.1.4 Outputs
The new candidates will be available to vote for during the election.
3.1.11.2 External Interfaces

3.1.11.2.1 User Interfaces
A GUI will be available to keep the user out of the system code, and to allow simplicity of use.

3.1.11.2.2 Hardware Interfaces
Fresh databases should be created.

3.1.11.2.3 Software Interfaces
Election candidates should be available for the voting application.

3.1.12 Election Status Statistics

3.1.12.1 Specification

3.1.12.1.1 Introduction
During the election, administrators will have the option of extracting information from the database regarding the status or result of an election, specifically the total number of votes thus far and the number of votes for each candidate.

3.1.12.1.2 Inputs
Administrators should choose to see the election status from an administrative function menu.

3.1.12.1.3 Processing
Data from the database will be gathered and displayed to administration in a clear and thorough manner. Certain statistics made need to be calculated from the data before presentation.

3.1.12.1.4 Outputs
The data from the election should be presented clearly to the administrator. This data must include the total number of votes along with the number of votes for each candidate. It should indicate the leading candidates at each position.

Optional statistics that could be presented include the percentage of the vote a candidate received, vote totals within each school, and anything else the SA may be curious of.

3.1.12.2 External Interfaces

3.1.12.2.1 User Interfaces
A GUI will be setup to allow user navigation and choices of outputting and saving data.

3.1.12.2.4 Communication Interfaces
All of the information retrieved will be accessed directly from the appropriate databases. The GUI will get all of its information by having direct access to all of the available databases.

3.1.13 Confirmation Number Calculator

3.1.13.1 Specification

3.1.13.1.1 Introduction
When voters bring their receipts to verify their votes at the paper trail location, the administrators must check the data on the receipt against the data on the paper trail. When discrepancies arise, the confirmation number can help in determining whether the discrepancy is more likely to be due to a forging of the receipt or an actual error. To determine where the error is, the administrator may want to run the vote information on the receipt through the confirmation number calculator to obtain the confirmation number that would match.

3.1.13.1.2 Inputs
The calculator should accept the timestamp, unique id, and candidates voted for.

3.1.13.1.3 Processing
Through a process that makes the chance of a duplicate number sufficiently small, the calculator should result in a unique, but mathematically predictable, number based on the data given.

3.1.13.1.4 Outputs
The calculator should output the confirmation number, a number unique to the information inputted.

3.1.14 Administrator Access Edits

3.1.14.1 Specification

3.1.14.1.1 Introduction
Initially, the system should allow a selected few officials access to the administrator functions. In the event that administrator access needs to be added or removed, current administrators can edit the access logins that the system accepts. In order to decrease the likelihood of an access disagreement, all administrators will have to select to remove someone before that person is actually denied access to the system.
3.1.14.1.2 Inputs
Administrators can choose to add new access logins by clicking on an “add” button and entering a login name and password. They can remove access logins by clicking a “remove” button and selecting the administrator to remove.

3.1.14.1.3 Processing
The application should first check to see that the information to add a new administrator is correct and return the user to the input screen with an error message if not. If the information is correct, that login name and password should be given access to the system. If the user selects to remove a current administrator, the application should record that removal and check for removals from the other remaining administrators. If all other administrators have also selected the removal, the administrator who has been selected for removal should have their login name eliminated from the list of those granted access to the system. If not, then the application should simply record and remember that the current user has selected that removal.

3.1.14.1.4 Outputs
Access should now be granted to newly added logins and denied to newly removed ones.

3.1.14.2 External Interfaces

3.1.14.2.1 User Interfaces
The interface should initially include two buttons corresponding to the two options to add or remove administrators. If the user chooses “add”, then a new interface with boxes to input the relevant information should be displayed. If the user chooses “remove”, a list of administrators should be displayed and the user should be able to select one from the list to remove.

3.2 Performance Requirements
The system requires a high level of performance from each component. Specifically, there must be a guarantee that the vote information that students are confirming from the application is quickly and reliably reaching the main terminal to be counted and printed. Votes cannot be lost in the transmission of the data. This includes making sure that the main terminal can be run confidently, without fear of a crash. The server housing the online application must maintain a certain level of reliability since no votes will be able to come in during a period of downtime. Any more than 20 students having a problem due to the software would constitute this system as totally unreliable software. Thus, an acceptable error rate would be one for every 1000 votes that come into the system.

3.3 Attributes

3.3.1 Availability
The administrator will control the availability of the E-voting system. There will be a designated time period each day where the voting will be allowed to take place. The duration of the voting will be determined by Syracuse University.

3.3.2 Security

3.3.2.1 Encryption
Encryption will be important to maintain the integrity of our E-Voting application. Encryption will be used to mask the identity of the voters and also the candidates they voted for. To accomplish this, we will use public-key encryption. Public-key encryption uses a combination of a private key and a public key. The private key is known only to the host computer, while the public key is given by the host computer to any computer that wants to communicate securely with it. To decode an encrypted message, a computer must use the public key, provided by the originating computer, and its own private key. A popular public-key encryption utility is called Pretty Good Privacy (PGP), which will be used as part of the implementation.

3.3.2.2 Firewall
A firewall will be present within our E-Voting application. The purpose of the firewall is to protect against malicious packets from being sent to the server that may cause a denial of service. Any denial of service will prevent the voting process from moving forward. We plan to run our E-Voting application through the Syracuse University server. There should be a firewall already in place that will satisfy our needs. SSH is the protocol that we will use to allow users to log in and vote. The firewall will only accept requests through SSH. This will be effective because only Syracuse University students will be able to log in to the server.

3.3.2.3 Unique ID and Confirmation Number
The unique id and confirmation number are both vital elements of the verification process. The process by which a unique id number is assigned to each voter must be kept secret to the public so that numbers cannot be fabricated and used to harm the integrity of the election. The confirmation number calculator will use a function to produce the number based on the input information of the candidates voted for, time, and voter’s unique id. This function must also be kept from the public so that receipts cannot be altered and the paper trail verification can operate smoothly.

3.3.3 Maintainability
The maintenance of the E-voting system will be handled by the SA, though the system development team should be available to assist with any serious issues.

3.3.4 Transferability / conversion
The software will allow voting candidates to be changed on a yearly basis by the administrator.
3.5 Glossary

Encryption: The process of concealing information or data into a cipher or code, especially to prevent unauthorized access.

Firewall: Computing a part of a computer system or network that is designed to block unauthorized access while permitting outward communication.

Public-key encryption: A cryptographic system that uses two keys, a public key known to everyone and a private key known only to the recipient of the message.

S-Tunnel: A program that allows you to encrypt arbitrary TCP connections inside the secure sockets layer.

3.6 High Level Design
See “eVotingHighLevelDesign.ppt”, contained in this directory, for an overview of our high level design.

See “High Level Design Outline.doc”, contained in this directory, for a text version outline of our high level design.

See “eVotingFlowChart.vsd”, contained in this directory, for a flow chart of our projected system performance.